REMARKS

Applicants respectfully request the Examiner to reconsider and again examine the claims in view of the following remarks.

Claims 1-43 are pending in the application. Claims 1-43 are rejected. Claims 9, 16, 17, 20, 21, 24, 25, 32, 33, 36, 38, and 39 are amended herein. Claims 1, 15, 23, 31, 37, and 41 are also amended herein but for reasons of clarity and not for reasons of patentability, as will be apparent.

A substitute specification is attached in accordance with the amended claims, in order to change the word "overriding" to "overridden" in most instances. Applicant submits that no new matter is introduced with the attached substitute specification.

FIGS. 5 and 6 are amended herein in accordance with amendments made in the substitute specification.

Before discussing below the particular rejections made by the Examiner, Applicant would like to provide an overview. The present invention is operable to store commands, which in some embodiments are display commands, for example scene graph display commands. The present invention can remove overridden, redundant, and/or superfluous commands before storing them from time to time as a so-called "dynamic snapshot," which is representative of a *system state*. In contrast, Applicants submit below that Trueblood, which is used by the Examiner, stores display information, but does not store a system state as display commands. Applicants further submit below that Deniau et al., which is also used by the Examiner, does not remove overridden, redundant, and/or superfluous commands. Now, turning to the specific rejections:

The Rejections under 35 U.S.C. §101

The Examiner rejects Claims 23-36 under 35 U.S.C. §101 as being directed to non-statutory subject matter.

With regard to Claims 23-36, which recite "[a] computer program medium having computer readable code thereon for storing commands, "the Examiner asserts that "Applicant discloses a … computer readable medium can include a readable memory device' and '[t]he computer readable medium can also include a communications link…having program code segments carried thereon." The Examiner further asserts that "[t]hese claims do not fall into one of the four statutory categories (process, machine, manufacture, or composition of matter) because the disclosure suggests that the program medium includes signals… ."

As an initial matter, Applicant submits that the passage from the specification reproduced by the Examiner, and appearing at page 28 of the substitute specification, actually reads in full: "[t]he computer readable medium can also include *a communications link, either optical, wired, or wireless*, having program code segments carried thereon as digital or analog signals. Contrary to the Examiner's assertion, Applicant submits that the definition provided by the Applicant describes a physical link, which is either optical, wired, or wireless, and is therefore, a composition of matter.

Applicant does not know if the Examiner recognizes the format presented in Claims 23-36 to constitute so-called Beauregard claims. Therefore, Applicant presents a discussion of Beauregard claims below for the Examiner's convenience.

Applicant respectfully submits that the format of Claims 23-36 represent statutory subject matter under In re Beauregard, 53 F.3d 1583, 35 USPQ2d 1383 (Fed. Cir. 1995).

According to Beauregard, Id., "[o]n August 4, 1994, the Board rejected Beauregard's computer program product claims on the basis of the printed matter doctrine. Beauregard appealed. The Commissioner now states 'that computer programs embodied in a tangible medium, such as floppy diskettes, are patentable subject matter under 35 U.S.C. § 101 and must be examined under 35 U.S.C. §§ 102 and 103.' The Commissioner states that he agrees with Beauregard's position on appeal that the printed matter doctrine is not applicable. Thus, the parties are in agreement that no case or controversy presently exists."

In accordance with Beauregard, the Manual of Patent Examining Procedure at §2106(a) states:

> ...a claimed **computer-readable medium** encoded with a data structure defines structural and functional interrelationships between the data structure and the computer software and hardware components which permit the data structure's functionality to be realized, and is thus statutory.
>
> Similarly, computer programs claimed as computer listings *per se*, i.e., the descriptions or expressions of the programs, are not physical "things." They are neither computer components nor statutory processes, as they are not "acts" being performed. Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed elements of a computer which permit the computer program's functionality to be realized. In contrast, a claimed **computer-readable medium** encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. Accordingly, it is important to distinguish claims that define descriptive material *per se* from claims that define statutory inventions. [emphasis added]

Thus, Applicant submits that Claims 23-36, which constitute Beauregard claims, are proper under 35 U.S.C. §101.

In view of the above, Applicant submits that the rejection of Claims 23-36 under 35 U.S.C. §101 should be removed.

<u>The Rejections under 35 U.S.C. §103(a)</u>

<u>Trueblood in View of Deniau et al.</u>

The Examiner rejects Claims 1 and 8-43 under 35 U.S.C. §103(a) as being unpatentable over Trueblood (U.S. Patent number 5,893,053) in view of Deniau et al. (U.S. Publication umber 2003/0222883). The Examiner asserts that Trueblood records "...a first set of commands....to a command queue....to provide a first dynamic snapshot...in a first system state." Applicant respectfully disagrees.

With regard to independent Claims 1 and 37, the Examiner recognizes that Trueblood fails to teach the claimed eliminating selected ones of [overridden], redundant, and superfluous commands from the command queue. The Examiner relies upon Deniau et al. as teaching this aspect. The Examiner concludes that "[i]t would have been obvious to one of ordinary skill in the art at the time the invention was made to update only objects that need to be updated by eliminating selected once of overridden, redundant, and superfluous commands." Applicant notes again that that the word "overriding" has been changed throughout the application to read "overridden," and respond accordingly as if the Examiner had used the word "overridden."

As the Examiner is aware, and as found in MPEP §2142, in order to establish a prima facie case of obviousness "...the prior art reference (or prior art references when combined) must teach or suggest all the claim limitations." Applicants respectfully submit that the Examiner has not met this burden in order to establish prima facie obviousness.

Applicant submits that Claim 1 is patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al., since the cited references neither describe nor suggest "... recording a first set of commands to a command queue to provide a first dynamic snapshot, wherein <u>*the first dynamic snapshot corresponds to a set of commands associated with a first system state*</u>; ... storing the first dynamic snapshot... ; recording one or more additional sets of commands to the command queue; ... <u>*eliminating selected ones of overridden, redundant, or superfluous commands from the command queue*</u> to provide a

second dynamic snapshot, wherein *the second dynamic snapshot corresponds to a set of commands associated with a second system state*; and storing the second dynamic snapshot ...,"as set forth in Claim 1.

With this particular arrangement, the present invention reduces the size required for storage of the dynamic snapshots, i.e., the system states. For example, referring to FIG. 3, at page 14, lines 2-8 of the substitute specification, it is described that:

> Shortly before the time that the next dynamic snapshot is stored in the storage device 36 (FIG. 1) the dynamic snapshot 122 is updated to a state then corresponding to the ATC display system 20 (FIG. 1). The dynamic snapshot is updated by appending the command stack 124 to the dynamic snapshot 122, to become the next dynamic snapshot. It should be understood that, without further processing, the dynamic snapshot 122 would progressively grow in size. Therefore, overridden, redundant, and/or superfluous commands can be removed from the command queue 120 to provide a dynamic snapshot 122 that is reduced in size.

Formation of the so-called dynamic snapshots and the removal of overridden, redundant, and/or superfluous display commands therefrom is further described throughout the specification, including in conjunction with FIG. 3. As described above, it will be understood that the removal of such overridden, redundant, and/or superfluous display commands from the dynamic snapshot (i.e., from the stored system state) can greatly reduce the number of stored commands in each stored system state, and therefore, the associated storage space required.

As described above, the Examiner asserts that Trueblood records "...a first set of commands....to a command queue....to provide a first dynamic snapshot...in a first system state." Applicant respectfully disagrees for the following reasons.

Applicant first again makes note that the reference designations 70 and 72 used in the specification of Trueblood are reversed in FIG. 3 and Applicant leaves the discrepancy uncorrected in the discussion below.

In contrast to the claimed arrangement, as shown in FIG. 3, Trueblood stores very large amounts of display information in an X-command file 68 (*in which commands are stored, but which is not representative of a system state*) and in a state file 72 (*in which data rather than commands is stored*). Trueblood also stores information in a control file 74 an in an event file 70. The storage of such large amounts of data results in a requirement for a much larger memory in which to store the data than required by the present invention. The storage of such large amounts of data also tends to result in a slower recall time of the data than for the present invention.

In particular, the X-command file 68 of Trueblood is used to store ***all*** display X-commands, which continually accumulate during operation, resulting in storage of a very large amount of information, *and which does not therefore, represent a system state*. For example, at column 5, lines 32-41, Trueblood describes:

> an X-server communication daemon 58 is interposed between the application programs 26, 28 and 30 and the X-Window system 50. The X-server communication daemon 58 intercepts all X-protocol commands exchanged between the X-Window system 50 and the client programs 26, 28 and 30. All such commands are copied to the state tracking client 64. State tracking client 64 time-stamps the stream of X-protocol commands and writes the stamped stream out to an X-command file 68 in a mass-storage device 24, such as a hard disk drive. [emphasis added]

The state file 72 of Trueblood is used to store *pixel* information (i.e., data) representative of a displays state, not commands representative of a system state as in the present invention. For example, at column 6, lines 20-24, Trueblood describes, "[p]articularly, the state file stores the create parameters for each of the pixmaps, windows, color maps, cursors, fonts, and properties active in the display." [emphasis added] Storage of pixel information associated with a display is known to require a very large amount of storage space. The state file of Trueblood does not store commands as claimed, but instead stores pixel data and other data.

The event file 70 of Trueblood is merely used to store cursor movements. For example, at column 7, lines 7-9, Trueblood describes, [a] separate software component, the event-tracking client 66, receives all cursor events and stores them in an event file 72.

The control file 74 of Trueblood is merely used to store user inputs. For example, at column 7, lines 39-47, Trueblood describes:

> The system of the present invention further includes an X-VCR record graphic user interface (GUI) client 60 and a control file 74. The X-VCR record GUI client 60 extracts information from user input, such as a file label, description (i.e., air sector ID, screen state storage intervals) and stores it in control file 74. The record GUI client 60 also extracts and stores in control file 74 work station data accessible through UNIX, such as start and stop time of the recording and work station ID. [emphasis added]

As described above, the Examiner relies upon Deniau et al. as teaching the claimed eliminating selected ones of overridden, redundant, and superfluous commands from the command queue.

Contrary to the Examiner's reliance upon Deniau et al., Applicant submits that Deniau et al. merely teaches updating in a screen display only those portions of the screen display that have changed since a previous screen display. As recognized by the Examiner, Deniau et al. provides an example of this arrangement in conjunction with FIG. 5, in which Deniau et al. identifies a rectangle 540 in which a displayed circle 520 has moved from a first display 510 to a second display 530 at a later time. In accordance with the identified rectangle 540, Deniau et al. updates only the portion of the display 530 in the rectangle 540, leaving a displayed circle 525 and a displayed rectangle 515 unchanged.

In updating only within the rectangular region 540, Deniau et al. need not remove any overridden, redundant, or superfluous commands from a command queue as recited in Claim 1 of the present application. Deniau et al., in conjunction with FIG. 5, describes only a circle 520 that moves from one position to another position from the time of a display 510 to the time of a

display 530. If, for example, display commands in Deniau et al. included commands to change the color of the circle 520 to blue, then to black, then to purple, the overridden commands for blue and black would not be removed by Deniau et al. from a command queue as claimed, resulting in a larger storage of display commands in Deniau et al. Applicants submit that the Examiner is attempting to expand Deniau et al. beyond that which Deniau et al. teaches.

Examples of overridden, redundant and superfluous display commands are given in the specification. For example, at page 22, beginning at line 19 of the substitute specification, it is described in conjunction with FIG. 6:

> … An overriding display command is essentially a display command that reverses or overrides an overridden display command in the command queue that was earlier issued. For example, an earlier issued command can move an image of an aircraft to the right, and an overriding command later issued can move the image of the aircraft to the left by an equal amount.
>
> …A redundant display command is a display command that accomplishes no additional function in view of an earlier issued display command in the command queue 120. For example, an earlier issued display command can specify that the color of an aircraft image is red, and a redundant command later issued can again specify that the image of the aircraft is red.
>
> …A superfluous display command is a display command that is not valid in the given context. For example, a display command that sets the color of an object and associated display image to white, when a default color associated with the object is white, has no purpose and is superfluous.

Deniau et al. fails to describe or suggest overridden, redundant, or superfluous display commands, or removing such commands from a command queue as claimed.

Furthermore, Applicant submits that, even if the invention of Trueblood were to be combined with the invention of Deniau et al, still the claimed invention would not result. Instead, the resulting arrangement would store display components as described above, which are not *display commands stored as a system state*. The resulting arrangement would update *pixel*

information as in Trueblood only in portions of a display in which a display object has changed from one display to another as in Deniau et al.

In view of the above, Applicant submits that Claim 1 is patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al.

Claims 8-22 depend from and thus include the limitations of Claim 1. Thus, Applicant submits that Claims 8-22 are patentably distinct over the cited reference at least for the reasons discussed above in conjunction with Claim 1.

Applicant submits that amended Claim 9 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "...the commands include *two-dimensional display commands* associated with a scene graph and associated with a graphical display and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*," as set forth in amended Claim 9.

Applicant submits that amended Claim 16 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "...the commands include display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*... ," as set forth in amended Claim 16.

Applicant submits that amended Claim 17 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "...the commands include two-dimensional display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*... ," as set forth in amended Claim 17.

Applicant submits that amended Claim 20 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*... ," as set forth in amended Claim 20.

Applicant submits that amended Claim 21 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include two-dimensional display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board* ... ," as set forth in amended Claim 21.

For substantially the same reasons described above in conjunction with Claim 1, Applicant submits that independent Claim 23 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... instructions for recording a first set of commands to a command queue to provide a first dynamic snapshot, wherein *the first dynamic snapshot corresponds to a set of commands associated with a first system state*...; instructions for storing the first dynamic snapshot...; instructions for recording one or more additional sets of commands to the command queue; ...*instructions for eliminating selected ones of overridden, redundant, or superfluous commands from the command queue* to provide a second dynamic snapshot, wherein *the second dynamic snapshot corresponds to a set of commands associated with a second system state*; and instructions for storing the second dynamic ... ," as set forth in Claim 23.

Claims 24-36 depend from and thus include the limitations of Claim 23. Thus, Applicant submits that Claims 24-36 are patentably distinct over the cited reference at least for the reasons discussed above in conjunction with Claim 23.

Applicant submits that amended Claim 24 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*," as set forth in amended Claim 24.

Applicant submits that amended Claim 25 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include two-dimensional display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*," as set forth in amended Claim 25.

For substantially the same reasons described above in conjunction with Claim 1, Applicant submits that Claim 31 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "...instructions for *eliminating selected ones of overridden, redundant, or superfluous commands from within the intermediate dynamic snapshot* ....," as set forth in Claim 31.

Applicant submits that amended Claim 32 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*... ," as set forth in amended Claim 32.

Applicant submits that amended Claim 33 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include two-dimensional display commands associated

with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board* ... ," as set forth in amended Claim 33.

Applicant submits that amended Claim 36 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include two-dimensional display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*... ," as set forth in amended Claim 36.

For substantially the same reasons described above in conjunction with Claim 1, Applicant submits that independent Claim 37 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... a dynamic snapshot generator coupled to the recording proxy for providing dynamic snapshots, wherein *each dynamic snapshot corresponds to a respective set of commands and each set of commands is associated with a system state*, wherein *the dynamic snapshot generator is adapted to eliminate selected ones of overridden, redundant, or superfluous commands from each one of the command sets*... ," as set forth in Claim 37.

Claims 38-43 depend from and thus include the limitations of Claim 37. Thus, Applicant submits that Claims 38-43 are patentably distinct over the cited reference at least for the reasons discussed above in conjunction with Claim 37.

Applicant submits that amended Claim 38 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*," as set forth in amended Claim 38.

Applicant submits that amended Claim 39 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... the commands include two-dimensional display commands associated with a scene graph and associated with a graphical display, *which commands are adapted for interpretation by a three dimensional (3D) graphics circuit board*," as set forth in amended Claim 39.

For substantially the same reasons described above in conjunction with Claim 1, Applicant submits that Claim 41 is further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... *a processor adapted to combine the commands in the command queue to eliminate selected ones of overridden, redundant, or superfluous commands in the command queue*... ," as set forth in Claim 41.

Applicant submits that Claim 41 is still further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... *a command queue having a command stack portion; and a dynamic snapshot portion*... ," as set forth in Claim 41.

Applicant submits that Claims 42 and 43 are further patentably distinct over Trueblood, whether taken alone or in combination with Deniau et al, since the cited references neither describe nor suggest "... *the command stack portion...and...the dynamic snapshot portion*... ," as set forth in Claims 42 and 43.

In view of the above, Applicant submits that the rejection of Claims 1 and 8-43 under 35 U.S.C. §103(a) should be removed.

Trueblood in View of Deniau et al and Burt et al.

The Examiner rejects Claims 2-7 under 35 U.S.C. §103(a) as being unpatentable over Trueblood in view of Deniau et al. and in view of Burt et al. (U.S. Patent number 5,649,032). The Examiner recognizes that Trueblood and Deniau et al. do not teach the claimed first and second interval values. The Examiner relies on Burt to teach the interval values.

Claims 2-7 depend from and thus include the limitations of Claim 1. Thus, Applicant submits that Claims 2-7 are patentably distinct over the cited reference at least for the reasons discussed above in conjunction with Claim 1.

In view of the above Amendments and Remarks, Applicants submit that Claims 1-43 and the entire case are in condition for allowance and should be sent to issue and such action is respectfully requested.

The Examiner is respectfully invited to telephone the undersigning attorney if there are any questions regarding this Response or this application.

The Assistant Commissioner is hereby authorized to charge payment of any additional fees associated with this communication or credit any overpayment to Deposit Account No. 500845, including but not limited to, any charges for extensions of time under 37 C.F.R. §1.136.

Respectfully submitted,

Dated: July 7, 2006

DALY, CROWLEY, MOFFORD & DURKEE, LLP

By:

Kermit Robinson
Reg. No. 48,734
Attorney for Applicant(s)
354A Turnpike Street - Suite 301A
Canton, MA 02021-2714
Tel.: (781) 401-9988, Ext. 24
Fax: (781) 401-9966
kr@dc-m.com

Attachments: see appendix

27612.doc

Appendix:

A substitute specification is attached in a version showing changes and in a clean version.

FIGS. 5 and 6 are attached both as Replacement Sheets and also as Annotated Sheets Showing Changes.

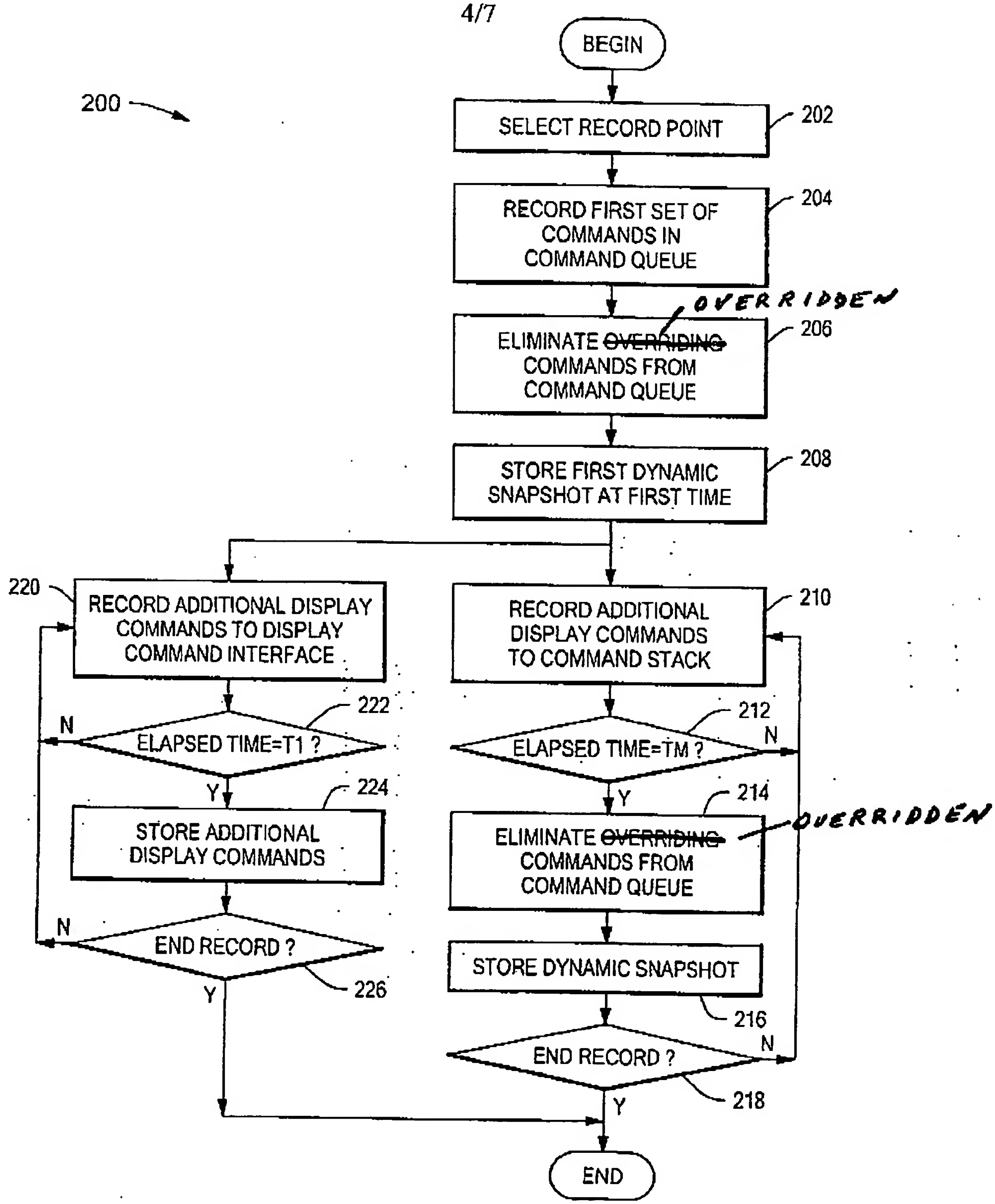
200
BEGIN
SELECT RECORD POINT
202
RECORD FIRST SET OF COMMANDS IN COMMAND QUEUE
204
OVERRIDDEN
ELIMINATE ~~OVERRIDING~~ COMMANDS FROM COMMAND QUEUE
206
STORE FIRST DYNAMIC SNAPSHOT AT FIRST TIME
208
220
RECORD ADDITIONAL DISPLAY COMMANDS TO DISPLAY COMMAND INTERFACE
222
ELAPSED TIME=T1 ?
N
Y
224
STORE ADDITIONAL DISPLAY COMMANDS
END RECORD ?
N
Y
226
RECORD ADDITIONAL DISPLAY COMMANDS TO COMMAND STACK
210
212
ELAPSED TIME=TM ?
N
Y
214
ELIMINATE ~~OVERRIDING~~ COMMANDS FROM COMMAND QUEUE
OVERRIDDEN
STORE DYNAMIC SNAPSHOT
216
END RECORD ?
N
Y
218
END

FIG. 5

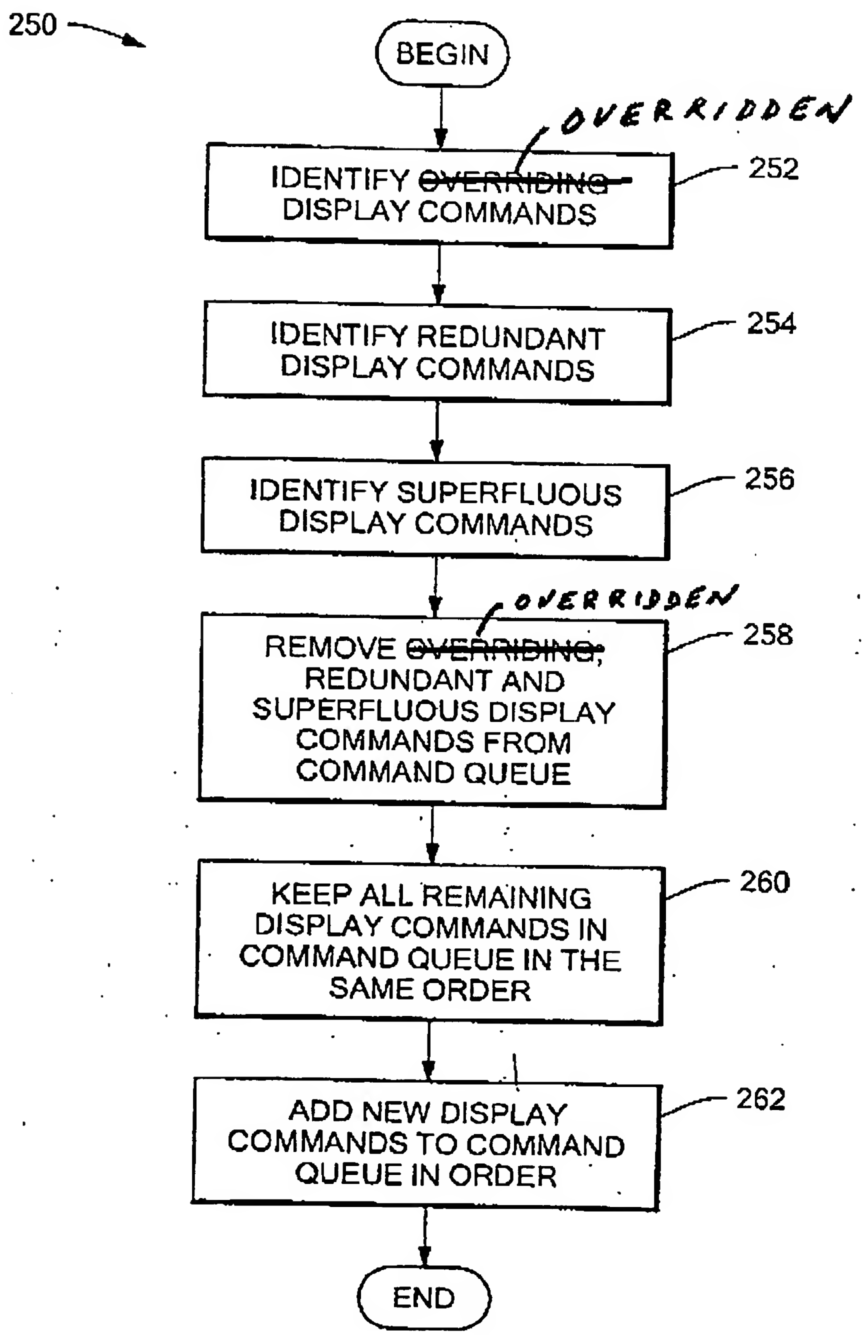
250
BEGIN
OVERRIDDEN
IDENTIFY ~~OVERRIDING~~ DISPLAY COMMANDS
252
IDENTIFY REDUNDANT DISPLAY COMMANDS
254
IDENTIFY SUPERFLUOUS DISPLAY COMMANDS
256
OVERRIDDEN
REMOVE ~~OVERRIDING,~~ REDUNDANT AND SUPERFLUOUS DISPLAY COMMANDS FROM COMMAND QUEUE
258
KEEP ALL REMAINING DISPLAY COMMANDS IN COMMAND QUEUE IN THE SAME ORDER
260
ADD NEW DISPLAY COMMANDS TO COMMAND QUEUE IN ORDER
262
END

FIG. 6

U.S. Patent Application No. 10/617,603

Filed On July 11, 2003

SUBSTITUTE SPECIFICATION

SHOWING CHANGES MADE

SYSTEM AND METHOD FOR ASYNCHRONOUS STORAGE AND PLAYBACK OF A SYSTEM STATE

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit under 35 U.S.C. §119(e) of U.S. Provisional Application No. 60/395,236 filed July 11, 2002 and U.S. Provisional Application No. 60/395,643 filed July 12, 2002, which applications are hereby incorporated herein by reference in their entirety.

STATEMENTS REGARDING FEDERALLY SPONSORED RESEARCH

Not applicable.

FIELD OF THE INVENTION

This invention relates generally to computer systems, such as air traffic control systems, and, more particularly, to a system and method for storing and playing back a state of a computer system.

BACKGROUND OF THE INVENTION

As is known in the art, air traffic control (ATC) is a service that promotes the safe, orderly, and expeditious flow of air traffic. Safety is principally a matter of preventing collisions with other aircraft, obstructions, and the ground, assisting aircraft in avoiding hazardous weather, assuring that aircraft do not operate in an airspace where operations are prohibited, and assisting aircraft in distress. Orderly and expeditious flow assures the efficiency of aircraft operations along the routes selected by the operator.

As is also known, air traffic control services are provided by air traffic control (ATC) systems. An air traffic control system is a type of computer and display system that processes data received from air surveillance radar systems for the detection and tracking of aircraft. Air traffic control systems are used for both civilian and military applications to determine the identity and locations of aircraft in a particular geographic area.

It is desirable in aircraft control systems to store data recorded, obtained, or otherwise provided by, the air traffic control system. In particular, it is desirable to be able to store and play back display data, including images shown on the display system of the air traffic control system. It is desirable to be able to record and play back air traffic control data in case of accident, for teaching purposes, for simulation purposes and the like.

Conventionally, there are several ways to record ATC display data. One technique is to use a video recorder to store ATC display data at each ATC display station. While this approach does not interrupt system operation (i.e., it has no impact on computer code or system performance and thus is a transparent recording technique), the video recorder approach has several drawbacks. For example, there is a relatively large amount of display data to store. If videotapes are used for storage, a relatively large number of videotapes are required to store all of the necessary display data associated with a period of time (e.g., 24 hours). Also, there are typically multiple ATC stations/terminals at which storage is desired. Thus, multiple video recorders or other recording devices are required. Furthermore, it is relatively time consuming to locate a specific location on a videotape during playback of the display data.

Another technique for transparently storing data is to provide computer code used in ATC systems having instructions for sending messages to a storage device, which identify to the storage device the operations being performed by the ATC system. The messages can be sent to the storage device either before or after a corresponding operation is performed. To playback what has occurred in the ATC system, the storage device sends the stored messages to the ATC system and the ATC display system carries out the messages.

However, if the developer of the computer code fails to include code to store a certain step or operation, or if the developer fails to include code to playback a certain message, then the record or playback will not be accurate. Another problem with this approach is that it is necessary to process a relatively large amount of data. Also, it is very expensive in terms of code development because it is very time consuming to include all of the additional computer code to store and play back the operations being displayed on the display system.

It would, therefore, be desirable to overcome the aforesaid and other disadvantages.

SUMMARY OF THE INVENTION

A system for asynchronous storage and playback of a system state provides "instantaneous images" of a system without interfering with the system behavior and response time. The images are associated with an ordered set of commands referred herein as a "dynamic snapshot." When stored and later executed in the proper order, the dynamic snapshot commands set the system to a snapshot state, which is representative of a time at which the dynamic snapshot was recorded.

In one particular embodiment, the system for asynchronous storage and playback of a system state includes a storage device for storing at certain times an image or "snapshot" of an air traffic control (ATC) display. The system can also store changes that occur on the display between the snapshots. With this particular arrangement, a system for transparently storing the ATC display is provided that does not interfere with other processing performed by the ATC system. By storing the display at certain times (i.e. storing the display snapshot), and then recording the changes on the display which occur between the certain times, the system records a reduced amount of data than is recorded in prior art approaches, without a decrease in the accuracy of the recording. While the system is described in association with an ATC display, it should be recognized that the system is applicable to any system having software commands or instructions.

In accordance with the present invention, a method of storing commands includes recording a first set of commands to a command queue as a first dynamic snapshot and storing the first dynamic snapshot. The method also includes recording one or more additional sets of commands to the command queue and eliminating ~~overriding~~overridden commands from the command queue to provide a second dynamic snapshot. The second dynamic snapshot is stored. In one embodiment, the commands are display commands associated with an ATC display.

In accordance with another aspect of the present invention, stored commands associated with a time of interest are played back by receiving the time of interest and retrieving a stored dynamic snapshot corresponding to a time at, or prior to, the time of interest. Additional sets of stored commands are also retrieved and appended to the dynamic snapshot to provide an intermediate dynamic snapshot. The intermediate dynamic snapshot is interpreted. In one particular embodiment, the stored dynamic snapshot includes display commands associated with an air traffic control (ATC) display and the interpreting results in a view of the ATC display corresponding to the time of interest. In one particular embodiment ~~overriding~~overridden commands within the intermediate dynamic snapshot are eliminated.

In accordance with another aspect of the present invention, a computer program medium for storing commands includes instructions for recording a first set of commands to a command queue as a first dynamic snapshot and instructions for storing the first dynamic snapshot. The computer program medium also includes instructions for recording one or more additional sets of commands to the command queue and instructions for eliminating ~~overriding~~overridden commands from the command queue to provide a second dynamic snapshot. The computer program medium also includes instructions for storing the second dynamic snapshot. In one embodiment, the commands are display commands associated with an ATC display.

In accordance with another aspect of the present invention, a computer program medium for storing commands includes instructions for receiving a time of interest and instructions for retrieving a stored dynamic snapshot corresponding to a time at, or prior to, the time of interest. The computer program medium also includes instructions for retrieving additional sets of stored commands and instructions for appending the additional sets of stored commands to the dynamic snapshot to provide an intermediate dynamic snapshot. The computer program medium also includes instructions for interpreting the intermediate dynamic snapshot. In one particular embodiment, the stored dynamic snapshot includes display commands associated with an air traffic control (ATC) display and the commands for

interpreting result in a view of the ATC display corresponding to the time of interest. In one particular embodiment, the computer program medium includes instructions for eliminating ~~overriding~~overridden commands from within the intermediate dynamic snapshot.

In accordance with another aspect of the present invention, a system includes a recording proxy, a dynamic snapshot generator, a command interface, and a storage device, all coupled so as to store dynamic snapshots and additional display commands in the storage device.

With this invention, the dynamic snapshot can be recorded and stored without interrupting real-time system operation. In one embodiment, this approach allows a display image to be stored without an impact on the user, i.e., without freezing a real-time display and also without burdening a computer code developer by requiring the developer to write additional computer code to record and/or store system commands. The present invention allows the asynchronous storage and playback of a stored system state (snapshot) without impacting system real-time operation.

The present invention finds application in display recorders (e.g., of the type used in air traffic control systems) as well as in systems (e.g., database systems) in which it is desirable to be able to rapidly recreate a data set without interfering with real-time system behavior (e.g., response time).

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing features of this invention, as well as the invention itself, may be more fully understood from the following description of the drawings in which:

FIG. 1 is a block diagram of an air traffic control (ATC) recording system, which utilizes a dynamic snapshot and has store and playback capability;

FIG. 2 is a diagram illustrating formation of a dynamic snapshot;

FIG. 3 is a block diagram of a dynamic snapshot command queue;

FIG. 4 is a diagram illustrating storage of dynamic snapshots and storage of intermediate sets of commands;

FIG. 5 is a flow diagram illustrating a set of processing steps to generate and to store a dynamic snapshot;

FIG. 6 is a flow diagram illustrating a set of processing steps to eliminate ~~overriding~~overridden commands from a dynamic snapshot;

FIG. 7 is a flow diagram illustrating a set of processing steps used to provide a playback of stored dynamic snapshots and stored intermediate sets of commands; and

FIG. 8 is a flow diagram illustrating an alternate set of processing steps used to provide a playback of stored dynamic snapshots and stored intermediate sets of commands.

DETAILED DESCRIPTION OF THE INVENTION

Before describing a system and method for asynchronous storage of a system snapshot, some introductory terms are described. As used herein, the terms "data storage" and "recording" are used synonymously to refer to retention of data. However, as used herein, the term "storage" can, in some embodiments, refer to retention of data for a particularly long period of time, for example, hours or days.

A scene graph will be understood to be a particular representation containing information about the geometry and appearance of objects appearing on a graphical display. The scene graph is a dynamic data structure within a computer program that can also be saved as a file. The scene graph includes data that describes shape objects (geometry and appearance), geometric structure relationships (geometric transformations, ordering, and grouping), global objects (how all shape objects are viewed, e.g., viewpoints, lights, backgrounds), behaviors (procedures for modifying information stored in a scene graph), and the like.

The scene graph is object oriented, having software objects that describe the shape objects and software commands that perform the behaviors upon the shape objects. For example, a scene graph can include a software object associated with an aircraft image, and a

scene graph display command can operate on the aircraft object to render the aircraft image on a graphical display.

The objects of a scene graph are created using software commands, for example a "create" command. The objects of a scene graph are operated upon using other commands, for example a "render" command, which causes an object to appear as an image on a video screen. Therefore, the scene graph, including the objects, is associated with a set of scene graph display commands.

A scene graph can be represented diagrammatically as a tree structure having "nodes" and interconnecting lines or "arcs." The scene graph data structure described above underlies the tree structure representation.

As used herein, the term "scene graph" is used to describe the underlying data structure associated with a scene graph, as opposed to the set of scene graph display commands or the scene graph tree structure.

It should be understood that a scene graph can be associated with more scene graph display commands than actually are used to generate images on a graphical display. For example, a scene graph can be associated with a set of "create" commands that represent scene graph objects, and not every object necessarily has a corresponding "render" command that generates an image on the graphical display.

Various high-level software application programmer interfaces (APIs) have been established to create a scene graph when presented with the scene graph display commands. For example Java3D and VRML provide high-level software to generate a scene graph. Lower level APIs have also been provided, including Open GL, and Direct 3D.

Scene graph techniques are conventionally used to provide a scene graph associated with three-dimensional images on a graphical display, for example in computer games. To this

end, hardware manufacturers have provided three dimensional (3D) graphics circuit boards, having local processing capability on the graphical circuit board, and having the ability to interpret scene graph data and providing a corresponding graphical display on a monitor.

Scene graph programming techniques, in conjunction with the 3D graphic circuit board, provide the ability to rapidly render a 3D image on a graphical display with minimal impact on a central computer processor. Images on the graphical display can also be rapidly updated with one or more display commands, interpreted by an API, and sent to the 3D graphics circuit board.

While existing scene graph APIs provide three-dimensional (3D) graphical objects and corresponding 3D images on a graphical display, a conventional air traffic control (ATC) display provides two-dimensional (2D) images. Two-dimensional images are conventionally provided without use of scene graphs and without taking advantage of the local processing capability of the 3D graphics circuit board. Instead, 2D images are conventionally generated with APIs that can interpret a very low-level "paint" command. A single paint command is able to render a simple shape, for example a line. Therefore, in order to render more complex shapes on an ATC display, such as aircraft and geographic features, numerous paint commands are conventionally generated, which are then interpreted by a low level API in conjunction with an ATC central processor. Therefore, the conventional ATC display requires substantial processing time provided by the ATC central processor, in order to process the large number of "paint" commands.

However, a 2D scene graph technique suitable for use in an ATC display system is described in U.S. Patent Application entitled "Scene Graph Based Display for Desktop Applications," filed on July 11, 2003, and assigned Application No. 10/617,599,[~~10/~~_______________, which is assigned to the assignee of the present invention and incorporated herein by reference. With the 2D scene graph technique, the ATC display images can be associated with two-dimensional (2D) scene graph display commands, which are

interpreted by a 2D scene graph API to provide a 2D scene graph. The 2D ATC display can be updated with 2D scene graph display commands.

As used herein, the term "snapshot" or "dynamic snapshot" should be broadly construed to refer to a set of system parameters describing a system "state" at a particular time. In one particular embodiment, the snapshot corresponds to a complete set of all system parameters describing the system state at the particular time. For example, in one particular embodiment, the dynamic snapshot is a snapshot of display commands representative of all images on a graphical display. It should be recognized that, whereas the exemplary dynamic snapshot includes the display commands themselves, the scene graph instead includes scene graph data generated by an API in response to scene graph display commands.

While the present invention is described in terms of scene graphs and scene graph commands below, and in particular in relation to an ATC display using 2D scene graphs, it should be appreciated that the present invention is also applicable to graphical display images using conventional "paint" commands and to systems other than ATC display systems.

Referring now to FIG. 1, a system 10 for asynchronous recording of a system snapshot includes a radar system 12. The radar system 12 provides radar information to a software agent 14. The radar information can include, for example, range, elevation, and azimuth information associated with one or more aircraft. The radar information can also include, for example, geographic information such as land topology, including mountains and hills.

The software agent 14 interprets the radar information and provides display commands 16. In one particular embodiment, the display commands are scene graph display commands including commands representative of objects and of renderings. A recoding proxy 18 forwards the display commands 16 to a real-time ATC display system 20 with no noticeable delay.

A user interface 15 allows a user to provide inputs to the software agent 14, for example

as mouse scrolls, mouse clicks, or the like, for example, to scroll, to zoom, and to otherwise interact with images presented on a monitor 26. The user actions are interpreted by the software agent 14 to generate additional display commands 16.

A real-time ATC display system 20 includes a central processing unit (CPU) 23 operating in conjunction with an API 21. The display commands 16 are received by the API 21, which operates upon the commands to generate a scene graph 24 stored within a graphics module 22. In one particular embodiment, the graphics module 22 is a three-dimensional (3D) graphics circuit board having a local processor (not shown) and capable of storing the scene graph 24. The graphics module 22 is coupled to a monitor 26, which provides an ATC display associated with the scene graph 22 and with the display commands 16.

It should be recognized that the system 10 has relatively few display commands using the scene graph techniques. In another embodiment using the "paint" display commands, the system 10 has more display commands. Therefore, the central processing unit (CPU) 23 of the real-time display system 20, spends less time on processing the scene graph display commands 16 than it would in processing "paint " display commands.

The recording proxy 18 also provides the display commands 16 to a dynamic snapshot generator 30 and to a display command interfuse 28. The dynamic snap shot generator 30 captures a system snapshot, i.e., a dynamic snapshot, associated with the real-time display system 20. In so doing, the dynamic snapshot generator 30 captures display commands associated with images presented on the ATC monitor 26 at a particular time. In one particular embodiment, the dynamic snapshot generator 30 captures display commands associated with all such images. In capturing the dynamic snapshot, the recording proxy 18 may append a time stamp to one or more particular display commands. Use of the time stamps will become apparent in association with FIGS. 7 and 8 below. In operation, the dynamic snapshot can be minimized in size using techniques described in conjunction with FIG. 3 below.

For reasons described above, the dynamic snapshot generated by the dynamic snapshot

generator 30 is not the same as the scene graph 24. The scene graph 24 contains processed display commands (processed by API 21) and the dynamic snapshot contains unprocessed display commands 16. The display commands included in a particular dynamic snapshot are representative of those objects and renderings appearing as images on the monitor 26 at a particular time.

The recording proxy 18 also provides the display commands 16 to the display command interface 28. The display command interface 28 captures one or more display commands 16 as they are provided by the software agent 14. Generally, the set of commands captured by the display command interface 28 is a relatively small set of display commands, which does not represent an entire dynamic snapshot. Instead, the set of commands captured by the display command interface 28, if presented to the real-time display system 20, would provide an update of one or more existing ATC display images.

A dynamic snapshot 32 provided by the dynamic snapshot generator 30 is stored by a storage device 36. Subsequent dynamic snapshots 32 are stored to the storage device 36 at first predetermined intervals thereafter. A set of display commands 34 provided by the display command interface 28 is also stored to the storage device 36. Subsequent sets of display commands 34 are stored to the storage device 36 at second predetermined intervals thereafter. The stored dynamic snapshots 32 and the stored sets of display commands 34 are herein referred to as "stored data." The sequence of storage to the storage device 36 is described in greater detail in conjunction with FIG. 4 below.

In one particular embodiment, the storage device 36 includes a tape media capable of storing digital data. In another embodiment, the storage device 36 is a solid-state storage device, for example a non-volatile solid-state storage device.

It should be appreciated that the dynamic snapshot generator 30 can provide the dynamic snapshot 32 to the storage device asynchronously from display activities of the real-time display system 20. It should also be appreciated that the display command interface 28

can provide the sets of display commands 34 to the storage device 36 asynchronously from the display activities of the real-time display system 20. Once recorded, the dynamic snapshots 32 and the sets of display commands 34 contain information necessary to later reconstruct images earlier seen on the monitor 26.

Upon playback, the recorded sets of display commands 38 are provided to a display command interface 42. Also, the recorded dynamic snapshots 40 are provided to a dynamic snapshot generator 44. During playback, in one embodiment, the dynamic snapshot generator 44 can provide a dynamic snapshot that is minimized in size as described below in conjunction with FIG. 3.

Two types of playback of the recorded data are described in conjunction with FIGS. 7 and 8 respectively. Let it suffice here to say that a recorded dynamic snapshot 48 and one or more recorded sets of display commands 46 are provided to a playback ATC display system 50 having an API 51 associated with a CPU 53, a graphics module 54, and a monitor 56. Upon playback, the dynamic snapshot provided by the dynamic snapshot generator 48, when interpreted by the API 51, provides a scene graph 54. The scene graph 54 is interpreted by the graphics module 52 to provide corresponding images on the monitor 56 , which, in one embodiment (e.g., FIG. 7), can correspond to a time of interest. However, in another embodiment (e.g., FIG. 8) , the scene graph 54 can be associated with a time before the time of interest and the sets of display commands 46 can bring the scene graph 54 forward in time to the time of interest.

It should be understood that the images thus presented on the monitor 56 correspond to those images seen at the earlier time of interest on the monitor 26, part of the real-time display system 26. Therefore, a user can view what was displayed earlier.

In operation, the display commands 16 received by the graphics module 22, provide the scene graph 24 as well as updates to the scene graph 24. As described above, a scene graph can be represented by a processed group of display commands and an update to the scene graph can

be represented by other processed display commands. For example, an image of a particular aircraft presented on the monitor 26 can correspond to one or more processed display commands that invoke an aircraft geometric object from the scene graph 22. Also, a change of characteristics of the particular aircraft, for example position, can correspond to one or more other processed display commands that change the invocation of the aircraft geometric object from the scene graph 22, while other geometric objects presented as images on the monitor 26 remain unchanged.

Referring now to FIG. 2, a first dynamic snapshot 100 is obtained at time t_0. The first dynamic snapshot 100 is combined with commands 102 recorded between time t_0 and some later time designated t_1 to provide a second dynamic snapshot 104 associated with the time t_1. The second dynamic snapshot 104 includes display commands associated with the time t_1, and therefore contains information about the state of images on the monitor 26 at the time t_1.

Using the inventive scene graph technique, a snapshot of the ATC display system can be represented using relatively few commands. The relatively few commands, however, correspond to images on the ATC display at a particular time. It is possible to "move" the dynamic snapshot forward through time by adding display commands, without affecting the operation of the system 10 (FIG. 1) in a substantive way.

It should be appreciated that retrieving a dynamic snapshot from the storage device 36 (FIG. 1) is an iterative process. That is, the process of retrieving the dynamic snapshot begins from a "known" system state, which is associated with a particular dynamic snapshot. The dynamic snapshot is then moved forward in time to a time of interest by adding subsequent recorded display commands to the dynamic snapshot.

Referring now to FIG. 3, an exemplary command queue 120 is contained in the dynamic snapshot generators 30, 44 (FIG. 1). The command queue 120 includes a dynamic snapshot portion 122 and a command accumulation portion, also referred to as a command stack 124. Once a recording of a particular dynamic snapshot 122 begins, commands issued

after that time are accumulated in the command stack 124 until the recording is completed, and until the time that a next dynamic snapshot is recorded.

Shortly before the time that the next dynamic snapshot is stored in the storage device 36 (FIG. 1) the dynamic snapshot 122 is updated to a state then corresponding to the ATC display system 20 (FIG. 1). The dynamic snapshot is updated by appending the command stack 124 to the dynamic snapshot 122, to become the next dynamic snapshot. It should be understood that, without further processing, the dynamic snapshot 122 would progressively grow in size. Therefore, ~~overriding~~overridden, redundant, and/or superfluous commands can be removed from the command queue 120 to provide a dynamic snapshot 122 that is reduced in size.

It is understood that in general, an overriding display command is a display command that reverses an action of an earlier issued overridden display command. A redundant display command is a display command that provides the same function as earlier issued display command. A superfluous display command is a display command that has no function in a given context. The removal of the ~~overriding~~overridden, redundant, and/or superfluous commands is also described in conjunction with FIG. 6.

In one embodiment, the system determines if any (and which) display commands within the command stack 124 override display commands within the dynamic snapshot 122. The ~~overriding~~overridden commands are removed from the dynamic snapshot 122 and from the command stack 124.

In order to remove ~~overriding~~overridden ~~commands~~ display commands, in one embodiment, display commands are translated or broken down into new sequences of display commands. The following show various examples of commands, which are broken down into new sequences.

The following are examples of two display commands:

group.add(index, element) // Inserts the specified element at the

```
                            // specified position in the group.
                            // Shifts the elements currently at that
                            // position (if any) and any subsequent
                            // element to the right (adds one to
                            // their indices).

group.remove(index)         // Removes the element at the specified
                            // position in this group. Shifts any
                            // subsequent element to the left
                            // (subtracts one from their indices).
```

It is not easy to determine if a remove display command overrides an add display command. Indeed, between add and remove display commands, other "insertions" and/or "removals" may have been performed and the elements could have shifted. Several options can resolve this uncertainty.

A first option converts "add" and "remove" display commands to sequences of "set" commands, e.g., group.set(element, index) display commands. For example:

group:

A	B	C	D	null	null
0	1	2	3	4	5

group.add(1, E):

```
group.set(4, D)
group.set(3, C)
group.set(2, B)
group.set(1, E)
```

group:

A	E	B	C	D	null
0	1	2	3	4	5

group:

A	B	C	D	null	null
0	1	2	3	4	5

group.remove(1):

```
group.set(1, C)
group.set(2, D)
group.set(3, null)
```

group:

A	C	D	null	null	null
0	1	2	3	4	5

With this particular option, two "set" commands override each other if and only if they are performed on the same group having the same index.

A second option breaks down the group into group nodes (linked list) and replaces add and remove commands with sequences of createGroupNode(), groupNode.setNext(groupNode),

groupNode.setElement(element) and groupNode.dispose() commands. For example:

group:

A	B	C	D
0	1	2	3

group.add(1, E):

```
n := createGroupNode()
n.setNext(1)
n.setElement(E)
0.setNext(n)
```

group:

A	E	B	C	D
0	n	1	2	3

group:

A	B	C	D
0	1	2	3

group.remove(1):

```
0.setNext(2)
1.dispose()
```

group:

A	C	D
0	2	3

The second option is usually faster than the first option described above during recording but requires more processing than the first option during playback (to recreate the group from the group nodes).

In one particular embodiment, for each new command applied to the command stack 124, the command queue 120 is examined in order to remove not only the ~~overriding~~overridden commands, but also redundant and superfluous display commands. The steps performed in order to remove the ~~overriding~~overridden, redundant, and superfluous display commands are described in conjunction with FIG. 6.

In one embodiment, by removing ~~overriding~~overridden, redundant, and superfluous commands form the command queue 120, the size of the command queue 120 can be maintained at or near about one hundred kilobytes.

Referring now to FIG. 4, a chart 150 showing storage of display commands includes a time scale 164 having times and time intervals. A first set of display commands 152, corresponding to a dynamic snapshot, is stored at time t0. The dynamic snapshot 152 can correspond, for example, to the dynamic snapshot 122 of FIG. 3 (also 32, FIG. 1) and the

storing can be provided, for example, by the storage device 36 of FIG. 1. As described above, the dynamic snapshot 152 includes display commands associated with images on an ATC display at the time t0. There can be any number of display commands in the dynamic snapshot 152.

After a time interval T1, a set of display commands 154 is stored. The set of display commands 154 can correspond, for example, to the set of display commands in the display command interface 28 of FIG. 1, and the storing can be again provided, for example, by the storage device 36 of FIG. 1. The set of display commands 154 can have any number of display commands, however, the set of display commands 154 generally has fewer display commands than the dynamic snapshot 152.

After a further time interval T2, a set of display commands 156 is stored. The set of display commands 156 can correspond, for example, to the another set of display commands in the display command interface 28 of FIG. 1, and the storing can be again provided, for example, by the storage device 36 of FIG. 1. The set of display commands 156 can have any number of display commands, including a different number of display commands than the set of display commands 154. However, the set of display commands 156 generally has fewer display commands than the dynamic snapshot 152.

Other sets of display commands, for example a set of display commands 160, are similarly stored. The time intervals T1, T2 and other time intervals associated with other ones of the sets of display commands, e.g., 154, 156, 158 can be the same time intervals or, in other embodiments, can be different time intervals.

At a time t1, another dynamic snapshot 160 is stored. The dynamic snapshot 160 can correspond, for example, to another of the dynamic snapshots 122 of FIG 3 (32, FIG. 1) and the storing can be provided, for example, by the storage device 36 of FIG. 1. As described above, the dynamic snapshot 160 includes commands associated with images on an ATC display at the time t1. There can be any number of display commands in the dynamic snapshot 160, and the

dynamic snapshot 160 need not have the same number of display commands as the dynamic snapshot 152.

The time interval between the time t1 and the time t0 is a time interval TM, which is larger than any one of the time intervals T1-TN. Therefore, the dynamic snapshots, for example, the dynamic snapshots 152, 160, are stored from time to time and the sets of display commands 156, 158, 160 are stored more often.

It should be appreciated that given the stored dynamic snapshot 152, and given the stored sets of display commands 156, 158, 160, which together form the stored data within the storage device 36 of FIG. 1, during a playback of the stored data, the stored dynamic snapshot 152 can essentially be moved forward in time by appending one of more of the stored sets of display commands 154, 156, 158 to the dynamic snapshot 152 to provide a more recent dynamic snapshot. Also, in one particular embodiment, upon appending the one or more sets of display commands 154, 156, 158 to the dynamic snapshot 152, ~~overriding~~overridden, redundant, and/or superfluous display commands can be removed from the resulting dynamic snapshot in much the same way as described above for recording in conjunction with FIG. 3.

In one particular embodiment, the time intervals T1, T2, TN are equal in duration and less than one second. In another embodiment, the time intervals T1, T2, TN are equal and less than five seconds. In another embodiment, the time intervals T1, T2, TN are equal and less than sixty seconds. However, in other embodiments, time intervals T1, T2, TN correspond to other times and may or may not be equal.

In one particular embodiment, the time interval TM is equal to time intervals between storage of others of the dynamic snapshots (not shown) and is greater than sixty seconds. In another embodiment, the time interval TM is greater than five minutes. In another embodiment, the time interval TM is greater than ten minutes. However, in other embodiments, time intervals TM corresponds to other times and may or may not be equal to time intervals between recording of other dynamic snapshots.

It should be appreciated that longer intervals between recording of dynamic snapshots results in less recording bandwidth, but a longer seeking time upon playback to find a time of interest. Shorter time intervals, therefore, result in greater recording bandwidth and shorter seeking times.

In one embodiment the techniques of the present invention have been used to demonstrate that: (1) recording does not significantly impact rendering performance (less than 2% of ATC central processor usage) and can be performed by background threads; (2) record/playback bandwidth is relatively low (the sets of display commands 154, 156, 158 represent infrequent changes to the dynamic snapshots, e.g. 152, 160, which are taken, for example, every 5 minutes); and (3) seeking a time of interest during a playback, i.e., to move a dynamic snapshot forward in time, can be relatively fast (less than 1 second for up to 5 minutes of time movement).

FIGS. 5-8 show process steps used to record and playback dynamic snapshots and sets of display commands, for example the dynamic snapshots 152, 160 and the sets of display commands 154, 156, 160 of FIG. 4.

Rectangular blocks of FIGS. 5-8 correspond to software processing steps associated with a software command or a set of software commands, and diamond blocks represent software decisions. Alternatively, the processing and decision blocks represent steps performed by functionally equivalent circuits such as a digital signal processor circuit or an application specific integrated circuit (ASIC). The flow diagrams do not depict the syntax of any particular programming language. Rather, the flow diagrams illustrate the functional information one of ordinary skill in the art requires to fabricate circuits or to generate computer software to perform the processing required of the particular apparatus. It should be noted that many routine program elements, such as initialization of loops and variables and the use of temporary variables are not shown. It will be appreciated by those of ordinary skill in the art

that unless otherwise indicated herein, the particular sequence of steps described is illustrative only and can be varied without departing from the spirit of the invention.

Referring now to FIG. 5, a flow chart 200 represents a series of steps used to record the dynamic snapshots 152, 160 and the sets of display commands 154, 156, 160 of FIG. 4. Processing begins at step 202, at which a point in time is selected at which recording will begin (referred to as a record point). The record point is typically associated with a recorded time stamp. The record point is a point at which at least a set of display commands corresponding to a dynamic snapshot has been acquired in the command queue, for example, the command queue 120 of FIG. 3.

At step 204, a first set of display commands is acquired and temporarily recorded in a solid-state memory or the like. The first set of display commands, as described above, corresponds to at least a set of display commands corresponding to a dynamic snapshot.

~~Overriding~~Overridden commands are eliminated at step 206 from the command queue, for example the command queue 120 of FIG. 3, after which the process continues to step 208, where the first dynamic snapshot is stored, for example to the storage device 36 of FIG. 1. In an alternate embodiment, redundant and/or superfluous commands are also removed at step 202 (see FIG. 6). As this is the first dynamic snapshot, there may be no ~~overriding~~overridden, redundant, or superfluous commands in the command queue. At step 208, the dynamic snapshot then stored is not deleted from the command queue 120.

Additional display commands are acquired and recorded to the command stack at step 210, for example, to the command stack 124 of FIG. 3. It will be appreciated that some of the additional display commands can be recorded in the command stack during the time that the dynamic snapshot is being stored to the storage device at step 208. Therefore, the storage of the dynamic snapshot can be performed asynchronously from the acquisition of additional display commands at step 210. Furthermore, the storage of the dynamic snapshot at step 208 can occur asynchronously from other aspects of the display processing.

At step 212, a decision is made as to whether it is time to store another dynamic snapshot, i.e., whether a time interval TM has elapsed since storage of the last dynamic snapshot. Time intervals associated with storage of the dynamic snapshot are discussed above in conjunction with FIG. 4. When enough time has elapsed, the process proceeds to step 214.

~~Overriding~~<u>Overridden</u> commands are eliminated from the command queue at step 214. As described in conjunction with FIG. 3, the ~~overriding~~<u>overridden</u> commands can be eliminated, for example, from among the command stack 124 and the dynamic snapshot 122, to generate a new dynamic snapshot. In an alternate embodiment, redundant and/or superfluous commands are also removed at step 214 (see FIG. 6).

At step 216, the new dynamic snapshot is stored to the storage device 36 (FIG. 1), after which, at step 218, a decision is made as to whether a total storage time has elapsed. A total storage time can be associated with, for example, a full storage device 36. If the total storage time has not elapsed, then the process returns to step 210, where additional display commands are accumulated to provide and store yet another dynamic snapshot at step 216.

At step 220, the additional display commands are also acquired, i.e., recorded in a display command interface, for example, the display command interface 28 of FIG. 1.

A decision is made at step 222 as to whether it is time to store the additional display commands, i.e., whether a time interval T1 has elapsed. Time intervals associated with storage of the additional display commands are discussed above in conjunction with FIG. 4. When enough time has elapsed, the process proceeds to step 224, at which time the additional display commands, for example, those additional display commands accumulated in the display command interface 28 (FIG. 1), are stored in the storage device 36. Upon storage of the additional display commands at step 224, the additional display commands can be deleted from the display command interface 28.

Like the dynamic snapshots stored at steps 208, 216, it will be appreciated that some of the additional display commands can be recorded at step 220 in the display command interface 28 during the time that the additional display commands are being stored to the storage device at step 224. Therefore, the storage of the additional display commands can be performed asynchronously from the acquisition of additional display commands at step 220. Furthermore, the storage of the additional display commands at step 224 can occur asynchronously from other aspects of the display processing.

At step 226, a decision is made comparable to the decision of step 218 described above, where it is determined whether the recording should be terminated. If the recording is not terminated, the process returns to step 220, where the earlier additional display commands are erased and new additional display commands are recorded.

With the process 200, the dynamic snapshots are stored at steps 208 and 216 with a time interval TM, while the additional display commands are stored at step 224 with a time interval T1. Generally the time interval T1 is less than the time interval TM. In other words, the dynamic snapshots are stored with longer time intervals between dynamic snapshots and the additional display commands are stored with shorter time intervals between sets of the additional display commands. In an alternate embodiment, the time interval T1 and/or TM can be varied at each cycle beginning at the decisions 226, 228 respectively.

Referring now to FIG. 6, a process 250 for removing ~~overriding~~overridden, redundant, and/or superfluous display commands from the command queue (120, FIG. 3) begins at step 252 by identifying ~~overriding~~overridden commands. An overriding display command is essentially a display command that reverses or overrides ~~a~~ an overridden display command in the command queue that was earlier issued. For example, an earlier issued command can move an image of an aircraft to the right, and an overriding command later issued can move the image of the aircraft to the left by an equal amount. ~~Overriding~~Overridden commands are also discussed in conjunction with FIG. 3 above.

At step 254, redundant display commands in the command queue 120 are identified. A redundant display command is a display command that accomplishes no additional function in view of an earlier issued display command in the command queue 120. For example, an earlier issued display command can specify that the color of an aircraft image is red, and a redundant command later issued can again specify that the image of the aircraft is red.

Superfluous display commands in the command queue are identified at step 256. A superfluous display command is a display command that is not valid in the given context. For example, a display command that sets the color of an object and associated display image to white, when a default color associated with the object is white, has no purpose and is superfluous.

In one particular embodiment, the search for ~~overriding~~overridden, redundant, and/or superfluous display commands is accelerated by using hash tables (e.g., only commands acting on the same object can be ~~overriding~~overridden or redundant display commands).

At step 258, the ~~overriding~~overridden, redundant, and/or superfluous display commands are removed from the command queue 120. In this way, the command queue 120 is minimized in size. In one air traffic control system, the command queue size stabilizes at about 100 kilobytes (kB).

At step 260, the display commands remaining in the command queue 120 are kept in their original order. At step 262, it is indicated that new display commands received in the command queue are also kept in order of reception.

Referring now to FIG. 7, an exemplary playback process 300 includes receiving, at step 302, a request to play back an ATC display associated with a specified time of interest. For an ATC system, for example, the request can be to display what happened at a specific time (e.g. "show what happened at 12:03:00.").

Once the request is received, the system locates on the storage device (36, FIG. 1) a dynamic snapshot having a time stamp prior to the time of interest, and retrieves the dynamic

snapshot at step 304. As described above, the dynamic snapshot corresponds to a set of display commands representative of the state of images on the monitor 26 (FIG. 1), corresponding to the time prior to the time of interest. The time stamp preferably corresponds to the closest earlier time at which a dynamic snapshot was recorded, prior to the time of interest. Continuing with the above example, assuming that a dynamic snapshot is stored in the storage device 36 every five (5) minutes beginning on the hour, then the desired dynamic snapshot is the dynamic snapshot stored at 12:00:00, having a corresponding time stamp.

At step 306, additional display commands are retrieved from the storage device 36. As described above in conjunction with FIG. 3, the additional display commands 154, 156, 158 are stored to the storage device between storage of successive dynamic snapshots 152, 160. The retrieved additional display commands, having time stamps, represent a group of display commands that occurred from the time of dynamic snapshot retrieved at step 304 up until the time of interest identified at step 302.

It should be appreciated that the dynamic snapshot retrieved at step 304, in combination with the additional display commands retrieved at step 306, represent the state of the real-time ATC display system 20 at the time of interest. Therefore, the dynamic snapshot retrieved at step 304, in combination with the additional display commands retrieved at step 306, corresponds to an intermediate dynamic snapshot, which corresponds to the dynamic snapshot retrieved at step 304 moved forward in time, as described in conjunction with FIG. 2.

In one particular embodiment, it is also possible to reduce the number of display commands from among the retrieved dynamic snapshot and the retrieved additional display commands to remove ~~overriding~~overridden, redundant, and/or superfluous display commands in much the same way as described above for recording in conjunction with FIG. 6.

At step 310, a display representative of the time of interest and corresponding to the intermediate dynamic snapshot is generated on the playback ATC display system 50 (FIG. 1). In another embodiment, the display representative of the time of interest is generated instead, or

additionally, on the real-time ATC display system 20 (FIG. 1).

The exemplary playback method 300 of FIG. 7 essentially moves the dynamic snapshot forward in time and thus has an advantage of eliminating the need to play through data from the time of the retrieved dynamic snapshot to the time of interest. Instead, a display corresponding to the time of interest is provided.

It should be appreciated that the playback described by the method 300 can be asynchronous from other aspects of the system 10 (FIG. 1). For example, the playback can be provided without impacting the real-time ATC display system 20 (FIG. 1).

Referring now to FIG. 8, an alternate exemplary playback process 350 includes receiving at step 352 a request to play back an ATC display associated with a specified time of interest. For an ATC system, for example, the request can be to display what happened at a specific time (e.g. "show what happened at 12:03:00.")

Once the request is received, the system locates on the storage device (36, FIG. 1) a dynamic snapshot having a time stamp prior to the time of interest, and retrieves the dynamic snapshot at step 354. As described above, the dynamic snapshot corresponds to a set of the display commands corresponding to state of images on the monitor 26 at a time prior to the time of interest. The time stamp preferably corresponds to the closest earlier time at which a dynamic snapshot was recorded prior to the time of interest. Continuing with the above example, assuming that a dynamic snapshot is stored in the storage device 36 every five (5) minutes beginning on the hour, then the desired dynamic snapshot is the dynamic snapshot stored at 12:00:00, having a corresponding time stamp.

At step 356 a display associated with the dynamic snapshot retrieved at step 354 is generated, for example, on the playback ATC display system 50 of FIG. 1. It should be appreciated that the display thus generated does not correspond to the time of interest provided at step 352. In another embodiment, the display associated with the dynamic snapshot retrieved

at step 354 is generated instead, or additionally, on the real-time ATC display system 20.

In order to move the display generated at step 356 forward in time, additional display commands are retrieved from the storage device 36 at step 358. The retrieved additional display commands correspond to those stored additional commands that correspond to times between the time of associated with the dynamic snapshot retrieved at step 354 and the time of interest.

At step 360, the display generated at step 356 is played forward in time by applying the retrieved additional display commands. The display can be played forward either at a speed representing normal time progression, at a speed representing fast time progression, or a speed representing slow time progression.

Unlike the playback method of FIG. 7, the exemplary playback method of FIG. 8 allows a user to view a generated display corresponding to any time between the time associated with the dynamic snapshot retrieved at step 354 and the time of interest. In another embodiment, the user can also view a generated display corresponding to a time after the time of interest.

It should be appreciated that the playback described by the method 350 can be asynchronous from other aspects of the system 10 (FIG. 1). For example, the playback can be provided without impacting the real-time ATC display system 20 (FIG. 1).

While the asynchronous storage and playback of a system snapshot has been shown and described in association with 2D scene graph display commands and with an ATC display, it should be appreciated, that in other embodiments, the display commands can include, but are not limited to, 3D scene graph display commands, and conventional "paint" display commands associated with other types of graphical displays, or any combination of 2D, 3D and "paint" display commands.

Also, while the playback method of FIGS. 7 and 8 has been shown and described to be

a playback occurring some time later than the generation of a corresponding real-time display, it should be understood that the playback can occur with only a very short time between the playback and the actual real-time display. In this way, the asynchronous storage and playback capability can be used to provide a system snapshot that user can view almost at the same time as the real-time display, without impacting the real-time system operation.

It should be appreciated that the "dynamic snapshot" storage and playback technique of the present invention finds application in a number of different systems but is particularly well suited for object-oriented systems. However, any system that provides software commands or instructions, having ~~overriding~~overridden, redundant, or superfluous commands or instructions as described above, and for which such ~~overriding~~overridden, redundant, or superfluous commands can be eliminated so as to limit the size of a dynamic snapshot, is suitable for the above-described techniques. For example, the above-described techniques can also be applied to a database system. It will be recognized that a database software applicaiton has software commands, which are suitable for removing ~~overriding~~overridden, redundant, and superfluous commands in much the same way as described above in conjunction with FIGS. 3 and 6. For example, a "set" command, used to place a number at a memory location can be overridden at a later time by another "set" command that places a number at the same location. Therefore, the techniques of the present invention can provide storage of dynamic snapshots and playback of the system states in much the same was as described above

The "dynamic snapshot" technique has a number of advantages including but not limited to the following: (1) there is no need to serialize/de-serialize the objects themselves (only the commands); (2) the default object's state does not need to be recorded; (3) if additional commands are recorded and time-stamped, then it is possible to playback the modification to the system from any dynamic snapshot; and (4) it is possible to seek very quickly by re-creating the dynamic snapshot at any particular time.

Having described preferred embodiments of the invention it will now become apparent to those of ordinary skill in the art that other embodiments incorporating these concepts may be

used. Additionally, the software included as part of the invention may be embodied in a computer program product that includes a computer useable medium. For example, such a computer usable medium can include a readable memory device, such as a hard drive device, a CD-ROM, a DVD-ROM, or a computer diskette, having computer readable program code segments stored thereon. The computer readable medium can also include a communications link, either optical, wired, or wireless, having program code segments carried thereon as digital or analog signals. Accordingly, it is submitted that that the invention should not be limited to the described embodiments but rather should be limited only by the spirit and scope of the appended claims. All publications and references cited herein are expressly incorporated herein by reference in their entirety.

What is claimed is:

\\server01\client_files\Prolaw documents\RTN-170AUS\32806.doc